

[illegible]

```

10  # Read in the data from the file
11  data = read.csv("data.csv")
12  # Print out the first few rows of the data
13  print(head(data))
14  # Print out the number of rows and columns in the data
15  print(dim(data))
16  # Print out the names of the columns in the data
17  print(colnames(data))
18  # Print out the names of the rows in the data
19  print(rownames(data))
20  # Print out the mean of each column in the data
21  print(colMeans(data))
22  # Print out the standard deviation of each column in the data
23  print(colSDs(data))
24  # Print out the correlation matrix of the data
25  print(cor(data))
26  # Print out the principal components of the data
27  print(prcomp(data))
28  # Print out the singular value decomposition of the data
29  print(svd(data))
30  # Print out the eigenvalues and eigenvectors of the data
31  print(eigen(data))
32  # Print out the singular values and singular vectors of the data
33  print(svdvals(data))
34  # Print out the singular vectors of the data
35  print(svdvec(data))
36  # Print out the singular values and singular vectors of the data
37  print(svdvalsvec(data))
38  # Print out the singular values and singular vectors of the data
39  print(svdvalsvec(data))
40  # Print out the singular values and singular vectors of the data
41  print(svdvalsvec(data))
42  # Print out the singular values and singular vectors of the data
43  print(svdvalsvec(data))
44  # Print out the singular values and singular vectors of the data
45  print(svdvalsvec(data))
46  # Print out the singular values and singular vectors of the data
47  print(svdvalsvec(data))
48  # Print out the singular values and singular vectors of the data
49  print(svdvalsvec(data))
50  # Print out the singular values and singular vectors of the data
51  print(svdvalsvec(data))
52  # Print out the singular values and singular vectors of the data
53  print(svdvalsvec(data))
54  # Print out the singular values and singular vectors of the data
55  print(svdvalsvec(data))
56  # Print out the singular values and singular vectors of the data
57  print(svdvalsvec(data))
58  # Print out the singular values and singular vectors of the data
59  print(svdvalsvec(data))
60  # Print out the singular values and singular vectors of the data
61  print(svdvalsvec(data))
62  # Print out the singular values and singular vectors of the data
63  print(svdvalsvec(data))
64  # Print out the singular values and singular vectors of the data
65  print(svdvalsvec(data))
66  # Print out the singular values and singular vectors of the data
67  print(svdvalsvec(data))
68  # Print out the singular values and singular vectors of the data
69  print(svdvalsvec(data))
70  # Print out the singular values and singular vectors of the data
71  print(svdvalsvec(data))
72  # Print out the singular values and singular vectors of the data
73  print(svdvalsvec(data))
74  # Print out the singular values and singular vectors of the data
75  print(svdvalsvec(data))
76  # Print out the singular values and singular vectors of the data
77  print(svdvalsvec(data))
78  # Print out the singular values and singular vectors of the data
79  print(svdvalsvec(data))
80  # Print out the singular values and singular vectors of the data
81  print(svdvalsvec(data))
82  # Print out the singular values and singular vectors of the data
83  print(svdvalsvec(data))
84  # Print out the singular values and singular vectors of the data
85  print(svdvalsvec(data))
86  # Print out the singular values and singular vectors of the data
87  print(svdvalsvec(data))
88  # Print out the singular values and singular vectors of the data
89  print(svdvalsvec(data))
90  # Print out the singular values and singular vectors of the data
91  print(svdvalsvec(data))
92  # Print out the singular values and singular vectors of the data
93  print(svdvalsvec(data))
94  # Print out the singular values and singular vectors of the data
95  print(svdvalsvec(data))
96  # Print out the singular values and singular vectors of the data
97  print(svdvalsvec(data))
98  # Print out the singular values and singular vectors of the data
99  print(svdvalsvec(data))
100 # Print out the singular values and singular vectors of the data
    
```

$$10^3 \text{ Pa} \quad 20 \quad 30 \quad 40 \quad 50 \quad 60 \quad 70 \quad 80 \quad 90 \quad 100$$

```

WGETS... 0/10... 0/10... 0/10... 0/10... 0/10...
  1st... 0/10... 0/10... 0/10... 0/10... 0/10...
  2nd... 0/10... 0/10... 0/10... 0/10... 0/10...
  3rd... 0/10... 0/10... 0/10... 0/10... 0/10...
  4th... 0/10... 0/10... 0/10... 0/10... 0/10...
  5th... 0/10... 0/10... 0/10... 0/10... 0/10...

```

[illegible]

1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214, 2215, 2216, 2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227, 2228, 2229, 2230, 2231, 2232, 2233, 2234, 2235, 2236, 2237, 2238, 2239, 2240, 2241, 2242, 2243, 2244, 2245, 2246, 2247, 2248, 2249, 2250, 2251, 2252, 2253, 2254, 2255, 2256, 2257, 2258, 2259, 2260, 2261, 2262, 2263, 2264, 2265, 2266, 2267, 2268, 2269, 2270, 2271, 2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2280, 2281, 2282, 2283, 2284, 2285, 2286, 2287, 2288, 2289, 2290, 2291, 2292, 2293, 2294, 2295, 2296, 2297, 2298, 2299, 2300, 2301, 2302, 2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312, 2313, 2314, 2315, 2316, 2317, 2318, 2319, 2320, 2321, 2322, 2323, 2324, 2325, 2326, 2327, 2328, 2329, 2330, 2331, 2332, 2333, 2334, 2335, 2336, 2337, 2338, 2339, 2340, 2341, 2342, 2343, 2344, 2345, 2346, 2347, 2348, 2349, 2350, 2351, 2352, 2353, 2354, 2355, 2356, 2357, 2358, 2359, 2360, 2361, 2362, 2363, 2364, 2365, 2366, 2367, 2368, 2369, 2370, 2371, 2372, 2373, 2374, 2375, 2376, 2377, 2378, 2379, 2380, 2381, 2382, 2383, 2384, 2385, 2386, 2387, 2388, 2389, 2390, 2391, 2392, 2393, 2394, 2395, 2396, 2397, 2398, 2399, 2400, 2401, 2402, 2403, 2404, 2405, 2406, 2407, 2408, 2409, 2410, 2411, 2412, 2413, 2414, 2415, 2416, 2417, 2418, 2419, 2420, 2421, 2422, 2423, 2424, 2425, 2426, 2427, 2428, 2429, 2430, 2431, 2432, 2433, 2434, 2435, 2436, 2437, 2438, 2439, 2440, 2441, 2442, 2443, 2444, 2445, 2446, 2447, 2448, 2449, 2450, 2451, 2452, 2453, 2454, 2455, 2456, 2457, 2458, 2459, 2460, 2461, 2462, 2463, 2464, 2465, 2466, 2467, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2487, 2488, 2489, 2490, 2491, 2492, 2493, 2494, 2495, 2496, 2497, 2498, 2499, 2500, 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 2509, 2510, 2511, 2512, 2513, 2514, 2515, 2516, 2517, 2518, 2519, 2520, 2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544, 2545, 2546, 2547, 2548, 2549, 2550, 2551, 2552, 2553, 2554, 2555, 2556, 2557, 2558, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2567, 2568, 2569, 2570, 2571, 2572, 2573, 2574, 2575, 2576, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2592, 2593, 2594, 2595, 2596, 2597, 2598, 2599, 2600, 2601, 2602, 2603, 2604, 2605, 2606, 2607, 2608, 2609, 2610, 2611, 2612, 2613, 2614, 2615, 2616, 2617, 2618, 2619, 2620, 2621, 2622, 2623, 2624, 2625, 2626, 2627, 2628, 2629, 2630, 2631, 2632, 2633, 2634, 2635, 2636, 2637, 2638, 2639, 2640, 2641, 2642, 2643, 2644, 2645, 2646, 2647, 2648, 2649, 2650, 2651, 2652, 2653, 2654, 2655, 2656, 2657, 2658, 2659, 2660, 2661, 2662, 2663, 2664, 2665, 2666, 2667, 2668, 2669, 2670, 2671, 2672, 2673, 2674, 2675, 2676, 2677, 2678, 2679, 2680, 26

200 年 10 月 05 日 10:14 分 28.45 分
 10 月 05 日 10:14 分 28.45 分
 10 月 05 日 10:14 分 28.45 分

[illegible]
$$-2\pi i \int_{\partial D} \frac{f(z)}{z} dz = 2\pi i \int_D \frac{f'(z)}{z} dz = 2\pi i \int_D \frac{f'(z)}{z} dz$$
[illegible]


```

11. # Import the necessary packages
12. import numpy as np
13. import pandas as pd
14. import matplotlib.pyplot as plt
15. import seaborn as sns
16. from sklearn.preprocessing import StandardScaler
17. from sklearn.model_selection import train_test_split
18. from sklearn.metrics import accuracy_score, confusion_matrix, roc_auc_score
19. from sklearn.linear_model import LogisticRegression
20. from sklearn.svm import SVC
21. from sklearn.tree import DecisionTreeClassifier
22. from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
23. from sklearn.neighbors import KNeighborsClassifier
24. from sklearn.naive_bayes import GaussianNB
25. from sklearn.metrics import classification_report
26. # Load the dataset
27. data = pd.read_csv('data.csv')
28. # Check the shape of the dataset
29. print(data.shape)
30. # Check the first few rows of the dataset
31. print(data.head())
32. # Check the distribution of the target variable
33. print(data['target'].value_counts())
34. # Split the data into training and testing sets
35. X_train, X_test, y_train, y_test = train_test_split(data[['feature1', 'feature2', 'feature3', 'feature4', 'feature5', 'feature6', 'feature7', 'feature8', 'feature9', 'feature10', 'feature11', 'feature12', 'feature13', 'feature14', 'feature15', 'feature16', 'feature17', 'feature18', 'feature19', 'feature20', 'feature21', 'feature22', 'feature23', 'feature24', 'feature25', 'feature26', 'feature27', 'feature28', 'feature29', 'feature30', 'feature31', 'feature32', 'feature33', 'feature34', 'feature35', 'feature36', 'feature37', 'feature38', 'feature39', 'feature40', 'feature41', 'feature42', 'feature43', 'feature44', 'feature45', 'feature46', 'feature47', 'feature48', 'feature49', 'feature50', 'feature51', 'feature52', 'feature53', 'feature54', 'feature55', 'feature56', 'feature57', 'feature58', 'feature59', 'feature60', 'feature61', 'feature62', 'feature63', 'feature64', 'feature65', 'feature66', 'feature67', 'feature68', 'feature69', 'feature70', 'feature71', 'feature72', 'feature73', 'feature74', 'feature75', 'feature76', 'feature77', 'feature78', 'feature79', 'feature80', 'feature81', 'feature82', 'feature83', 'feature84', 'feature85', 'feature86', 'feature87', 'feature88', 'feature89', 'feature90', 'feature91', 'feature92', 'feature93', 'feature94', 'feature95', 'feature96', 'feature97', 'feature98', 'feature99', 'feature100'], data['target'], test_size=0.2, random_state=42)
36. # Standardize the features
37. scaler = StandardScaler()
38. X_train = scaler.fit_transform(X_train)
39. X_test = scaler.transform(X_test)
40. # Train the Logistic Regression model
41. lr = LogisticRegression()
42. lr.fit(X_train, y_train)
43. # Predict the target variable for the test set
44. y_pred_lr = lr.predict(X_test)
45. # Calculate the accuracy score for the Logistic Regression model
46. accuracy_lr = accuracy_score(y_test, y_pred_lr)
47. # Print the accuracy score for the Logistic Regression model
48. print('Accuracy Score for Logistic Regression: ', accuracy_lr)
49. # Train the SVM model
50. svm = SVC()
51. svm.fit(X_train, y_train)
52. # Predict the target variable for the test set
53. y_pred_svm = svm.predict(X_test)
54. # Calculate the accuracy score for the SVM model
55. accuracy_svm = accuracy_score(y_test, y_pred_svm)
56. # Print the accuracy score for the SVM model
57. print('Accuracy Score for SVM: ', accuracy_svm)
58. # Train the Decision Tree model
59. dt = DecisionTreeClassifier()
60. dt.fit(X_train, y_train)
61. # Predict the target variable for the test set
62. y_pred_dt = dt.predict(X_test)
63. # Calculate the accuracy score for the Decision Tree model
64. accuracy_dt = accuracy_score(y_test, y_pred_dt)
65. # Print the accuracy score for the Decision Tree model
66. print('Accuracy Score for Decision Tree: ', accuracy_dt)
67. # Train the Random Forest model
68. rf = RandomForestClassifier()
69. rf.fit(X_train, y_train)
70. # Predict the target variable for the test set
71. y_pred_rf = rf.predict(X_test)
72. # Calculate the accuracy score for the Random Forest model
73. accuracy_rf = accuracy_score(y_test, y_pred_rf)
74. # Print the accuracy score for the Random Forest model
75. print('Accuracy Score for Random Forest: ', accuracy_rf)
76. # Train the Gradient Boosting model
77. gb = GradientBoostingClassifier()
78. gb.fit(X_train, y_train)
79. # Predict the target variable for the test set
80. y_pred_gb = gb.predict(X_test)
81. # Calculate the accuracy score for the Gradient Boosting model
82. accuracy_gb = accuracy_score(y_test, y_pred_gb)
83. # Print the accuracy score for the Gradient Boosting model
84. print('Accuracy Score for Gradient Boosting: ', accuracy_gb)
85. # Train the K-Nearest Neighbors model
86. knn = KNeighborsClassifier()
87. knn.fit(X_train, y_train)
88. # Predict the target variable for the test set
89. y_pred_knn = knn.predict(X_test)
90. # Calculate the accuracy score for the K-Nearest Neighbors model
91. accuracy_knn = accuracy_score(y_test, y_pred_knn)
92. # Print the accuracy score for the K-Nearest Neighbors model
93. print('Accuracy Score for K-Nearest Neighbors: ', accuracy_knn)
94. # Train the Naive Bayes model
95. nb = GaussianNB()
96. nb.fit(X_train, y_train)
97. # Predict the target variable for the test set
98. y_pred_nb = nb.predict(X_test)
99. # Calculate the accuracy score for the Naive Bayes model
100. accuracy_nb = accuracy_score(y_test, y_pred_nb)
101. # Print the accuracy score for the Naive Bayes model
102. print('Accuracy Score for Naive Bayes: ', accuracy_nb)
103. # Print the classification report for the Logistic Regression model
104. print(classification_report(y_test, y_pred_lr))
105. # Print the classification report for the SVM model
106. print(classification_report(y_test, y_pred_svm))
107. # Print the classification report for the Decision Tree model
108. print(classification_report(y_test, y_pred_dt))
109. # Print the classification report for the Random Forest model
110. print(classification_report(y_test, y_pred_rf))
111. # Print the classification report for the Gradient Boosting model
112. print(classification_report(y_test, y_pred_gb))
113. # Print the classification report for the K-Nearest Neighbors model
114. print(classification_report(y_test, y_pred_knn))
115. # Print the classification report for the Naive Bayes model
116. print(classification_report(y_test, y_pred_nb))
117. # Plot the ROC curve for the Logistic Regression model
118. fpr_lr, tpr_lr = roc_curve(y_test, lr.predict_proba(X_test)[:, 1])
119. plt.plot(fpr_lr, tpr_lr, label='Logistic Regression')
120. # Plot the ROC curve for the SVM model
121. fpr_svm, tpr_svm = roc_curve(y_test, svm.predict_proba(X_test)[:, 1])
122. plt.plot(fpr_svm, tpr_svm, label='SVM')
123. # Plot the ROC curve for the Decision Tree model
124. fpr_dt, tpr_dt = roc_curve(y_test, dt.predict_proba(X_test)[:, 1])
125. plt.plot(fpr_dt, tpr_dt, label='Decision Tree')
126. # Plot the ROC curve for the Random Forest model
127. fpr_rf, tpr_rf = roc_curve(y_test, rf.predict_proba(X_test)[:, 1])
128. plt.plot(fpr_rf, tpr_rf, label='Random Forest')
129. # Plot the ROC curve for the Gradient Boosting model
130. fpr_gb, tpr_gb = roc_curve(y_test, gb.predict_proba(X_test)[:, 1])
131. plt.plot(fpr_gb, tpr_gb, label='Gradient Boosting')
132. # Plot the ROC curve for the K-Nearest Neighbors model
133. fpr_knn, tpr_knn = roc_curve(y_test, knn.predict_proba(X_test)[:, 1])
134. plt.plot(fpr_knn, tpr_knn, label='K-Nearest Neighbors')
135. # Plot the ROC curve for the Naive Bayes model
136. fpr_nb, tpr_nb = roc_curve(y_test, nb.predict_proba(X_test)[:, 1])
137. plt.plot(fpr_nb, tpr_nb, label='Naive Bayes')
138. # Print the ROC AUC score for the Logistic Regression model
139. roc_auc_lr = roc_auc_score(y_test, lr.predict_proba(X_test)[:, 1])
140. # Print the ROC AUC score for the SVM model
141. roc_auc_svm = roc_auc_score(y_test, svm.predict_proba(X_test)[:, 1])
142. # Print the ROC AUC score for the Decision Tree model
143. roc_auc_dt = roc_auc_score(y_test, dt.predict_proba(X_test)[:, 1])
144. # Print the ROC AUC score for the Random Forest model
145. roc_auc_rf = roc_auc_score(y_test, rf.predict_proba(X_test)[:, 1])
146. # Print the ROC AUC score for the Gradient Boosting model
147. roc_auc_gb = roc_auc_score(y_test, gb.predict_proba(X_test)[:, 1])
148. # Print the ROC AUC score for the K-Nearest Neighbors model
149. roc_auc_knn = roc_auc_score(y_test, knn.predict_proba(X_test)[:, 1])
150. # Print the ROC AUC score for the Naive Bayes model
151. roc_auc_nb = roc_auc_score(y_test, nb.predict_proba(X_test)[:, 1])
152. # Print the ROC AUC score for the Logistic Regression model
153. print('ROC AUC Score for Logistic Regression: ', roc_auc_lr)
154. # Print the ROC AUC score for the SVM model
155. print('ROC AUC Score for SVM: ', roc_auc_svm)
156. # Print the ROC AUC score for the Decision Tree model
157. print('ROC AUC Score for Decision Tree: ', roc_auc_dt)
158. # Print the ROC AUC score for the Random Forest model
159. print('ROC AUC Score for Random Forest: ', roc_auc_rf)
160. # Print the ROC AUC score for the Gradient Boosting model
161. print('ROC AUC Score for Gradient Boosting: ', roc_auc_gb)
162. # Print the ROC AUC score for the K-Nearest Neighbors model
163. print('ROC AUC Score for K-Nearest Neighbors: ', roc_auc_knn)
164. # Print the ROC AUC score for the Naive Bayes model
165. print('ROC AUC Score for Naive Bayes: ', roc_auc_nb)
166. # Plot the confusion matrix for the Logistic Regression model
167. cm_lr = confusion_matrix(y_test, y_pred_lr)
168. # Plot the confusion matrix for the SVM model
169. cm_svm = confusion_matrix(y_test, y_pred_svm)
170. # Plot the confusion matrix for the Decision Tree model
171. cm_dt = confusion_matrix(y_test, y_pred_dt)
172. # Plot the confusion matrix for the Random Forest model
173. cm_rf = confusion_matrix(y_test, y_pred_rf)
174. # Plot the confusion matrix for the Gradient Boosting model
175. cm_gb = confusion_matrix(y_test, y_pred_gb)
176. # Plot the confusion matrix for the K-Nearest Neighbors model
177. cm_knn = confusion_matrix(y_test, y_pred_knn)
178. # Plot the confusion matrix for the Naive Bayes model
179. cm_nb = confusion_matrix(y_test, y_pred_nb)
180. # Print the confusion matrix for the Logistic Regression model
181. print('Confusion Matrix for Logistic Regression: ', cm_lr)
182. # Print the confusion matrix for the SVM model
183. print('Confusion Matrix for SVM: ', cm_svm)
184. # Print the confusion matrix for the Decision Tree model
185. print('Confusion Matrix for Decision Tree: ', cm_dt)
186. # Print the confusion matrix for the Random Forest model
187. print('Confusion Matrix for Random Forest: ', cm_rf)
188. # Print the confusion matrix for the Gradient Boosting model
189. print('Confusion Matrix for Gradient Boosting: ', cm_gb)
190. # Print the confusion matrix for the K-Nearest Neighbors model
191. print('Confusion Matrix for K-Nearest Neighbors: ', cm_knn)
192. # Print the confusion matrix for the Naive Bayes model
193. print('Confusion Matrix for Naive Bayes: ', cm_nb)
194. # Print the overall accuracy score for the Logistic Regression model
195. print('Overall Accuracy Score for Logistic Regression: ', accuracy_lr)
196. # Print the overall accuracy score for the SVM model
197. print('Overall Accuracy Score for SVM: ', accuracy_svm)
198. # Print the overall accuracy score for the Decision Tree model
199. print('Overall Accuracy Score for Decision Tree: ', accuracy_dt)
200. # Print the overall accuracy score for the Random Forest model
201. print('Overall Accuracy Score for Random Forest: ', accuracy_rf)
202. # Print the overall accuracy score for the Gradient Boosting model
203. print('Overall Accuracy Score for Gradient Boosting: ', accuracy_gb)
204. # Print the overall accuracy score for the K-Nearest Neighbors model
205. print('Overall Accuracy Score for K-Nearest Neighbors: ', accuracy_knn)
206. # Print the overall accuracy score for the Naive Bayes model
207. print('Overall Accuracy Score for Naive Bayes: ', accuracy_nb)
208. # Print the overall accuracy score for the Logistic Regression model
209. print('Overall Accuracy Score for Logistic Regression: ', accuracy_lr)
210. # Print the overall accuracy score for the SVM model
211. print('Overall Accuracy Score for SVM: ', accuracy_svm)
212. # Print the overall accuracy score for the Decision Tree model
213. print('Overall Accuracy Score for Decision Tree: ', accuracy_dt)
214. # Print the overall accuracy score for the Random Forest model
215. print('Overall Accuracy Score for Random Forest: ', accuracy_rf)
216. # Print the overall accuracy score for the Gradient Boosting model
217. print('Overall Accuracy Score for Gradient Boosting: ', accuracy_gb)
218. # Print the overall accuracy score for the K-Nearest Neighbors model
219. print('Overall Accuracy Score for K-Nearest Neighbors: ', accuracy_knn)
220. # Print the overall accuracy score for the Naive Bayes model
221. print('Overall Accuracy Score for Naive Bayes: ', accuracy_nb)
222. # Print the overall accuracy score for the Logistic Regression model
223. print('Overall Accuracy Score for Logistic Regression: ', accuracy_lr)
224. # Print the overall accuracy score for the SVM model
225. print('Overall Accuracy Score for SVM: ', accuracy_svm)
226. # Print the overall accuracy score for the Decision Tree model
227. print('Overall Accuracy Score for Decision Tree: ', accuracy_dt)
228. # Print the overall accuracy score for the Random Forest model
229. print('Overall Accuracy Score for Random Forest: ', accuracy_rf)
230. # Print the overall accuracy score for the Gradient Boosting
```

Figure 3

100

242 34

[illegible]